Configuration management

A call for design patterns

@ at computing

Training

Consultancy

& Remote support

- Celebrating 30 years !
- One of NLUUG founders

maurice@atcomputing.nl

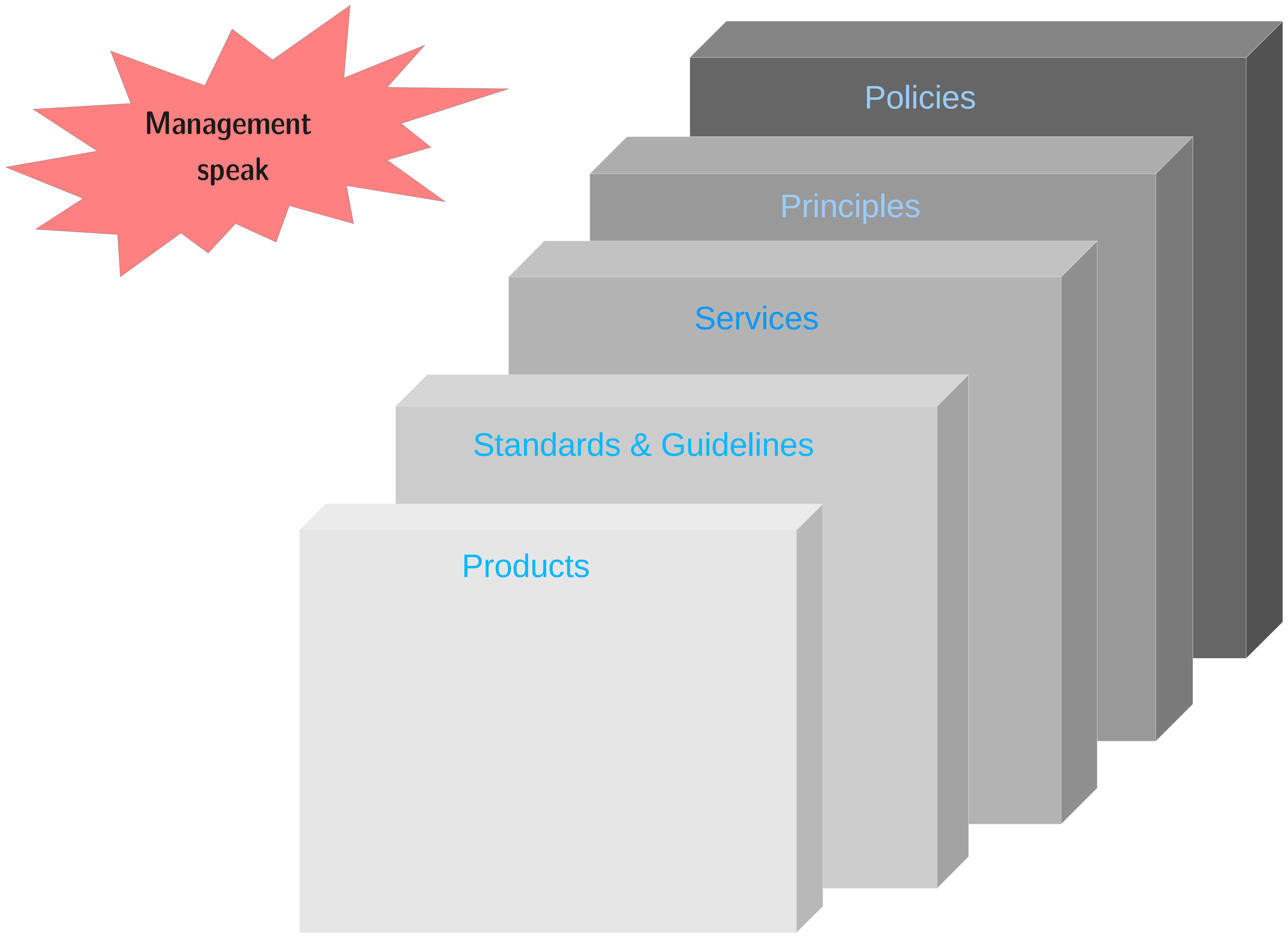
ing. Maurice Verheesen Msc.

- Technical account manager **AT Computing**
- Country & dutch team coordinator of the **FSFE**
- Electro engineer & Innovation Management
- Hobby == work

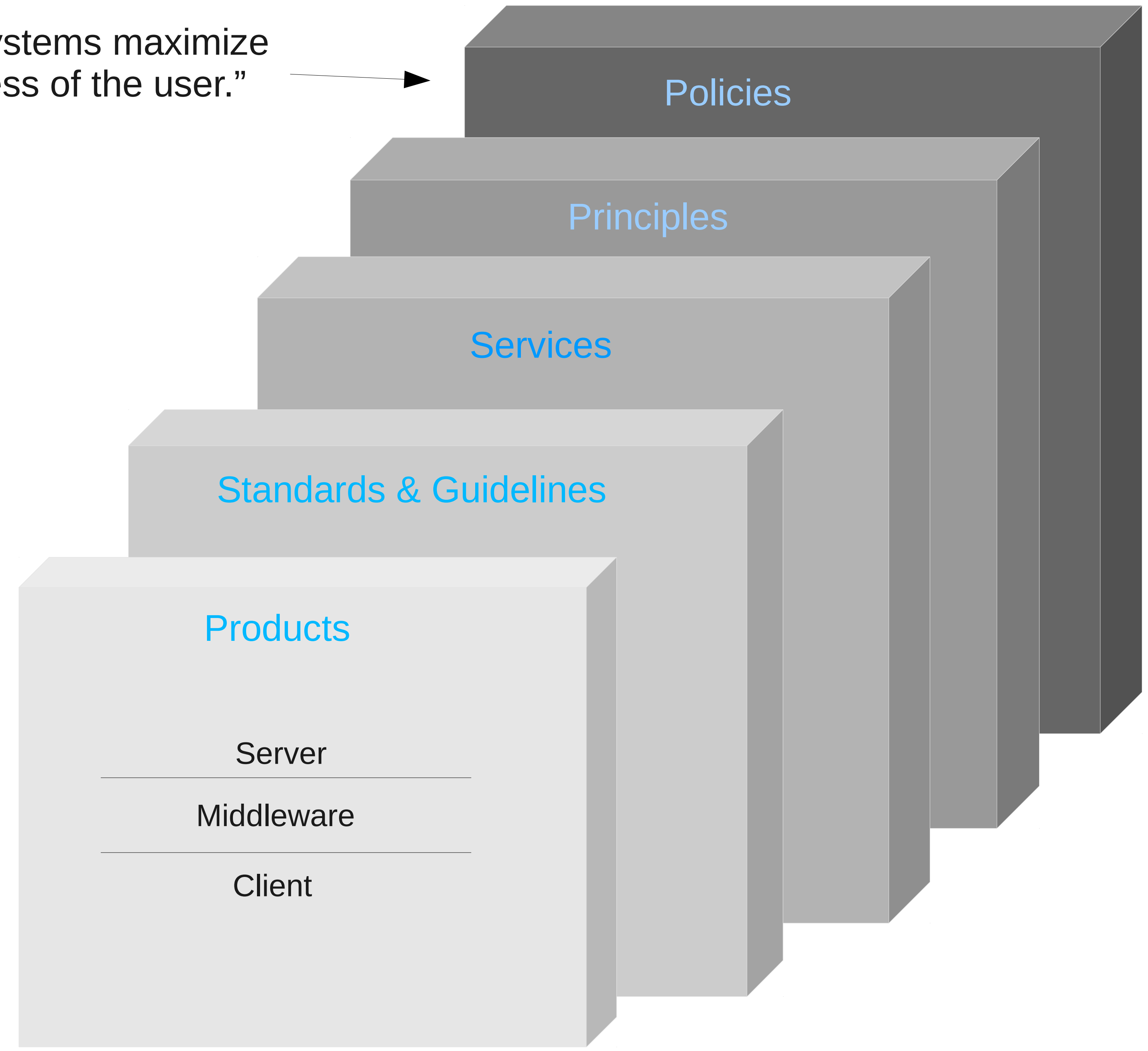
maurice@atcomputing.nl

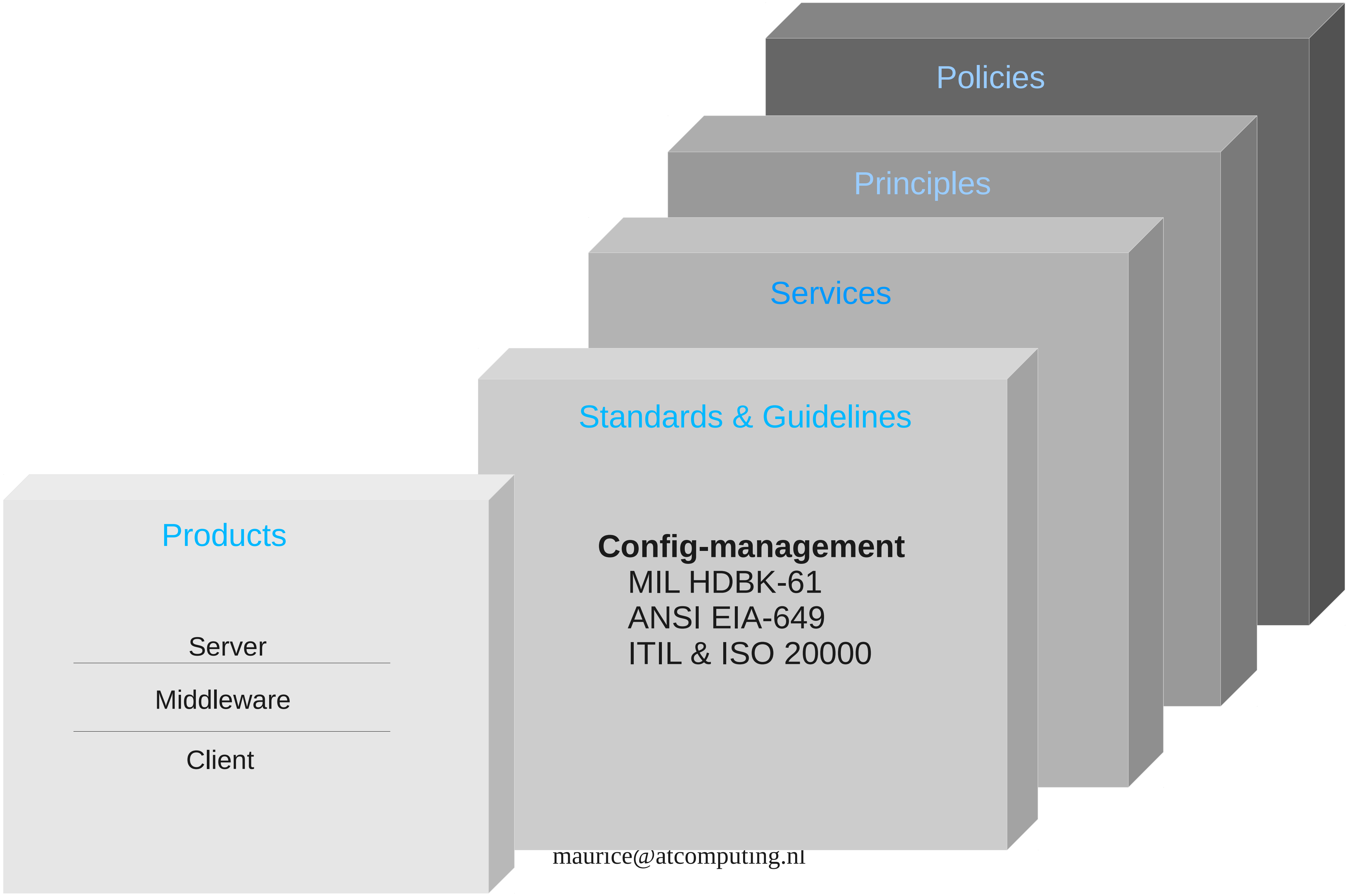
Contents

- Terms
- Tools
- Comparison of config-management tools
- Challenges
- Proposal : Design Patterns for CM


Management speak
Policies
Principles
Services
Standards & Guidelines
Products

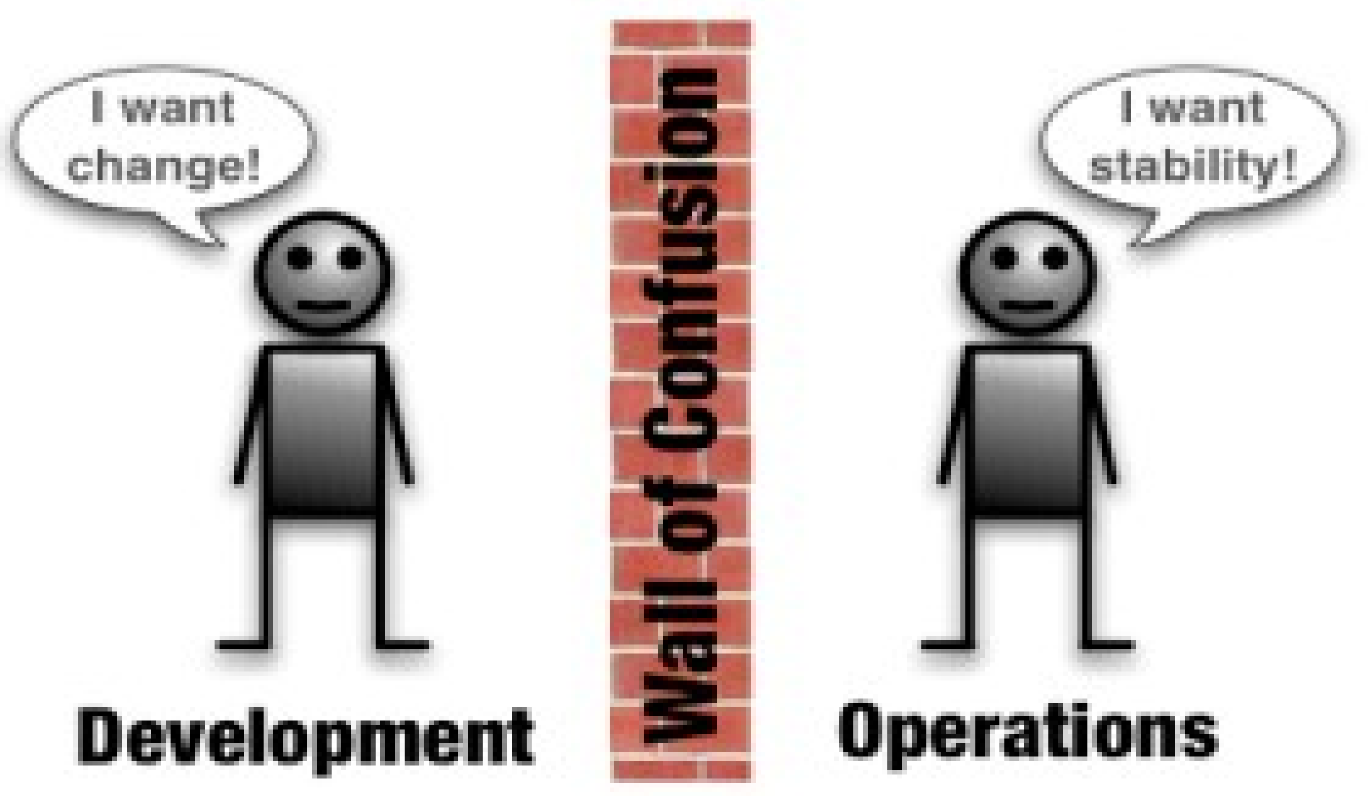
“Application systems maximize the effectiveness of the user.”

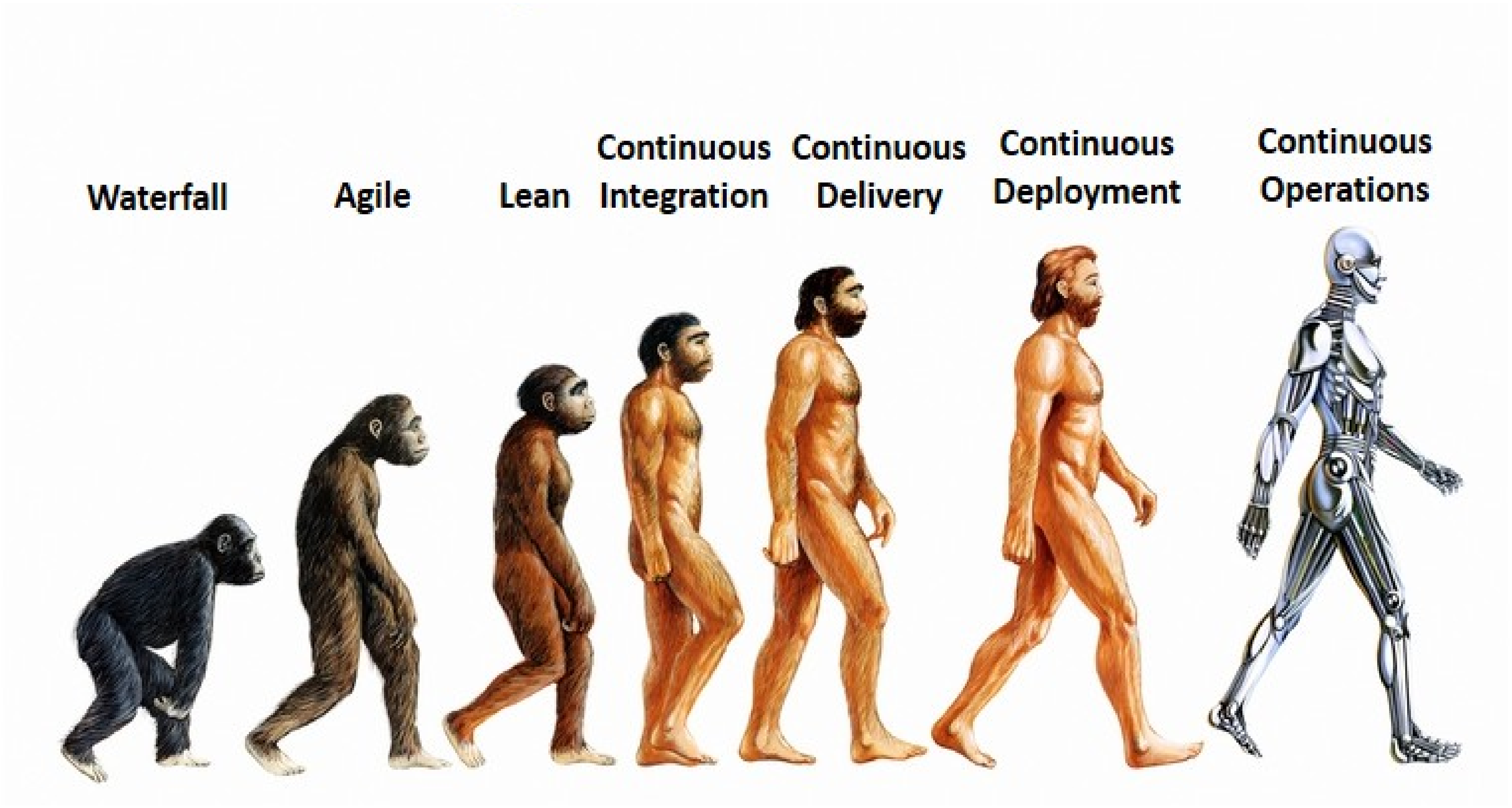




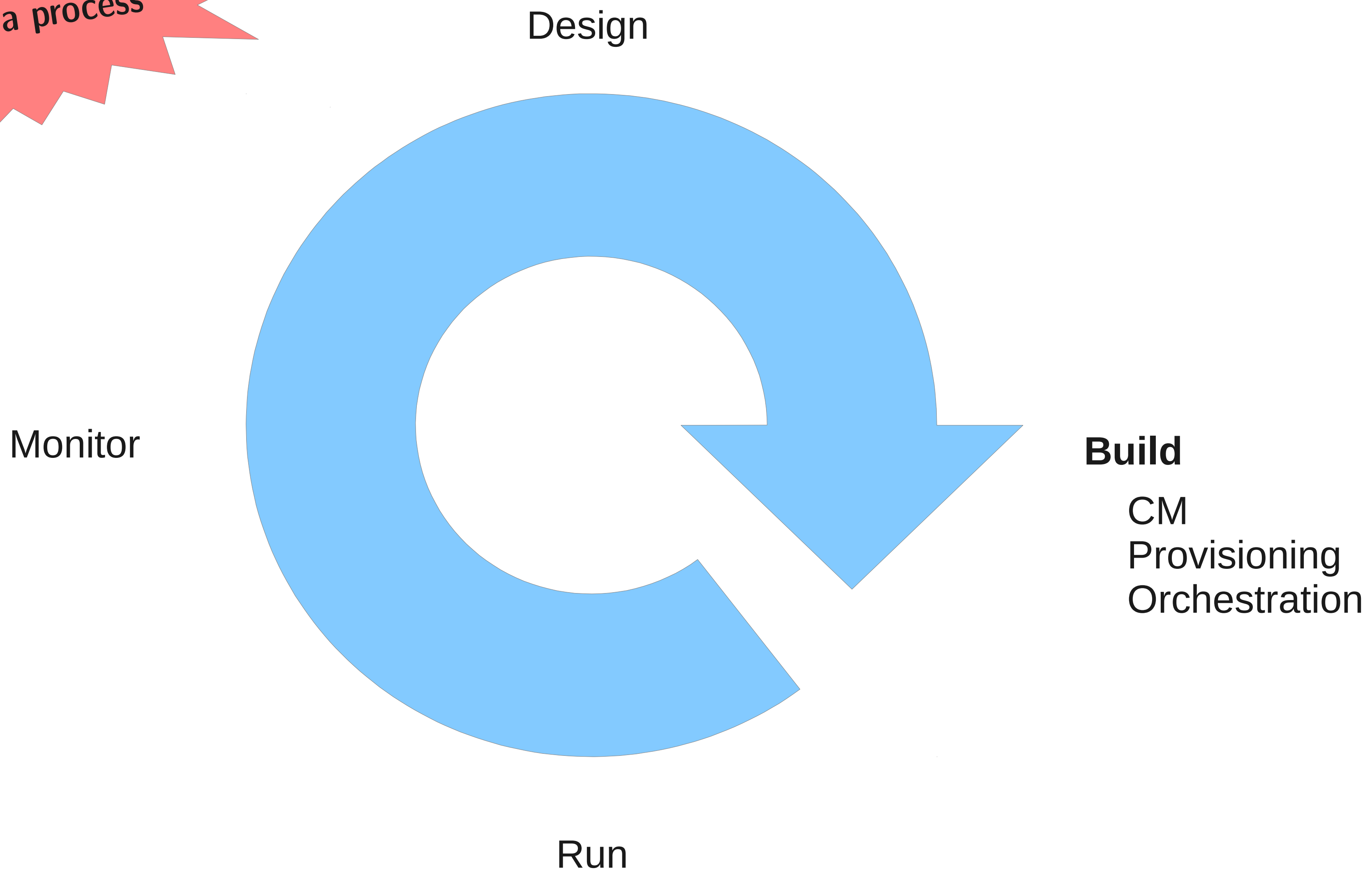
maurice@atcomputing.nl

Why are we doing this?

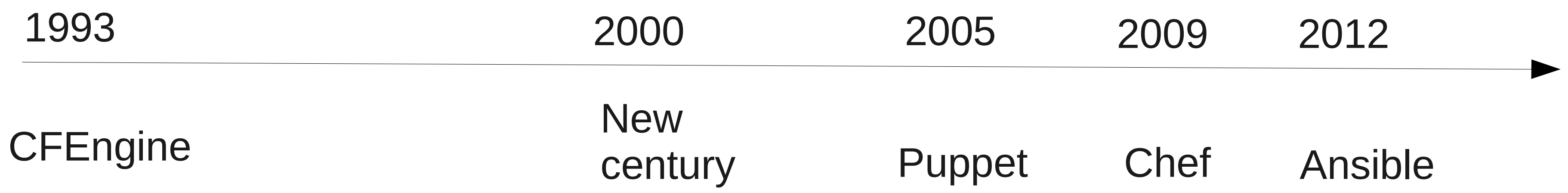

I want change!
Development
Wall of Confusion
I want stability!
Operations


Waterfall
Agile
Lean
Continuous Integration
Continuous Delivery
Continuous Deployment
Continuous Operations

Operations
as a process


Design
Build
CM
Provisioning
Orchestration
Run
Monitor

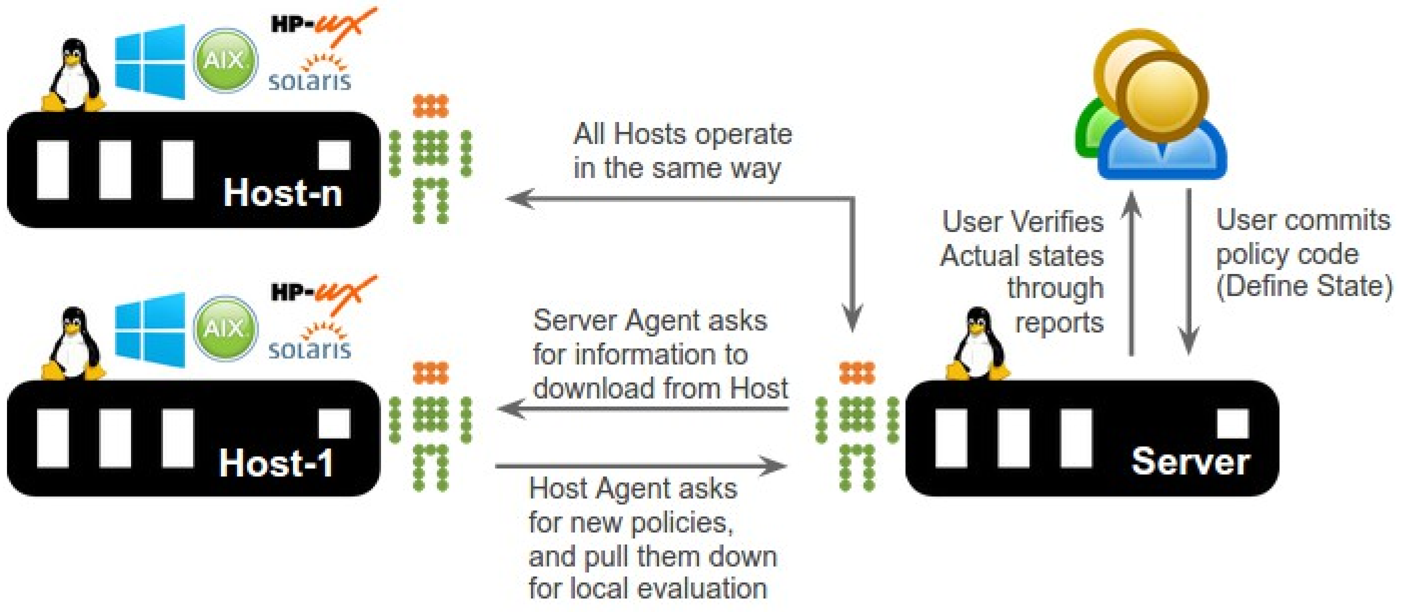
Tools


1993
CFEngine
2000
New century
2005
Puppet
2009
Chef
2012
Ansible

CFEngine

- Truly CM, the one that started it all
- Since v3 different DSL
- “... to define desired states of the IT infrastructure”
- “Lightweight **agents** continuously ensures that the actual states are converging toward the desired states, while reporting the outcome of each run“
- Promise theory
- Partial windows support

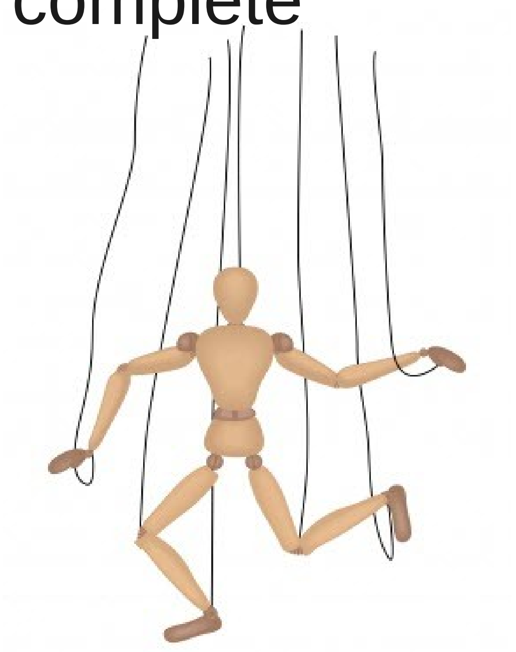
CFEngine


HP-ux
AIX
solaris
Host-n
Host-1
Server
All Hosts operate in the same way
Server Agent asks for information to download from Host
Host Agent asks for new policies, and pull them down for local evaluation
User Verifies Actual states through reports
User commits policy code (Define State)

```
body common control
{
  bundlesequence => { "my_test" };
}
bundle agent my_test{
 files:
  any::
   "/tmp/hello-world"
     create => "true";
}
```

Puppet

- CM, but also provisioning and orchestration these days
- Save CM values in database, instead of CM-files
- Generic modules, or roll your own (Puppetforge!)
- Ruby → Clojure (JVM)
- “... found that Puppet had the **biggest mind share** of the four products and represented the most complete picture for data center orchestration”
- Huge user base
- Windows



Puppet

The enterprise edition consists of:

- Puppet 3.8.0
- Puppet Server 1.0.9
- PuppetDB 2.3.2
- Facter 2.4.3
- Razor 1.0.0
- MCollective 2.7.0
- Hiera 1.3.4
- Dashboard 2.1.6

- Recipes
- Imperative !
- Cookbooks
- Ruby
- Agents
- Apache license
- Windows

- “New” kid on the block
- Focus op **orchestration**
- Python!
- Agentless
- “Impera-clarativish”
- Low learning curve
- Has things like “playbooks”, “roles”
- Windows

Comparison

	Agent	“manual” mode	Windows	RBAC	Multi-tenancy	Language	Focus	License
CFEngine	yes	local mode	partial	yes	no	c	CM	GPL
Puppet	yes	“yes”	yes	yes	yes	Ruby	CM + Pro + Orch	Apache
Chef	yes	yes	yes	yes	yes	Ruby	CM	Apache
Ansible	no	yes!	yes	yes	no	Python	Orch	GPL

Trouble in paradise

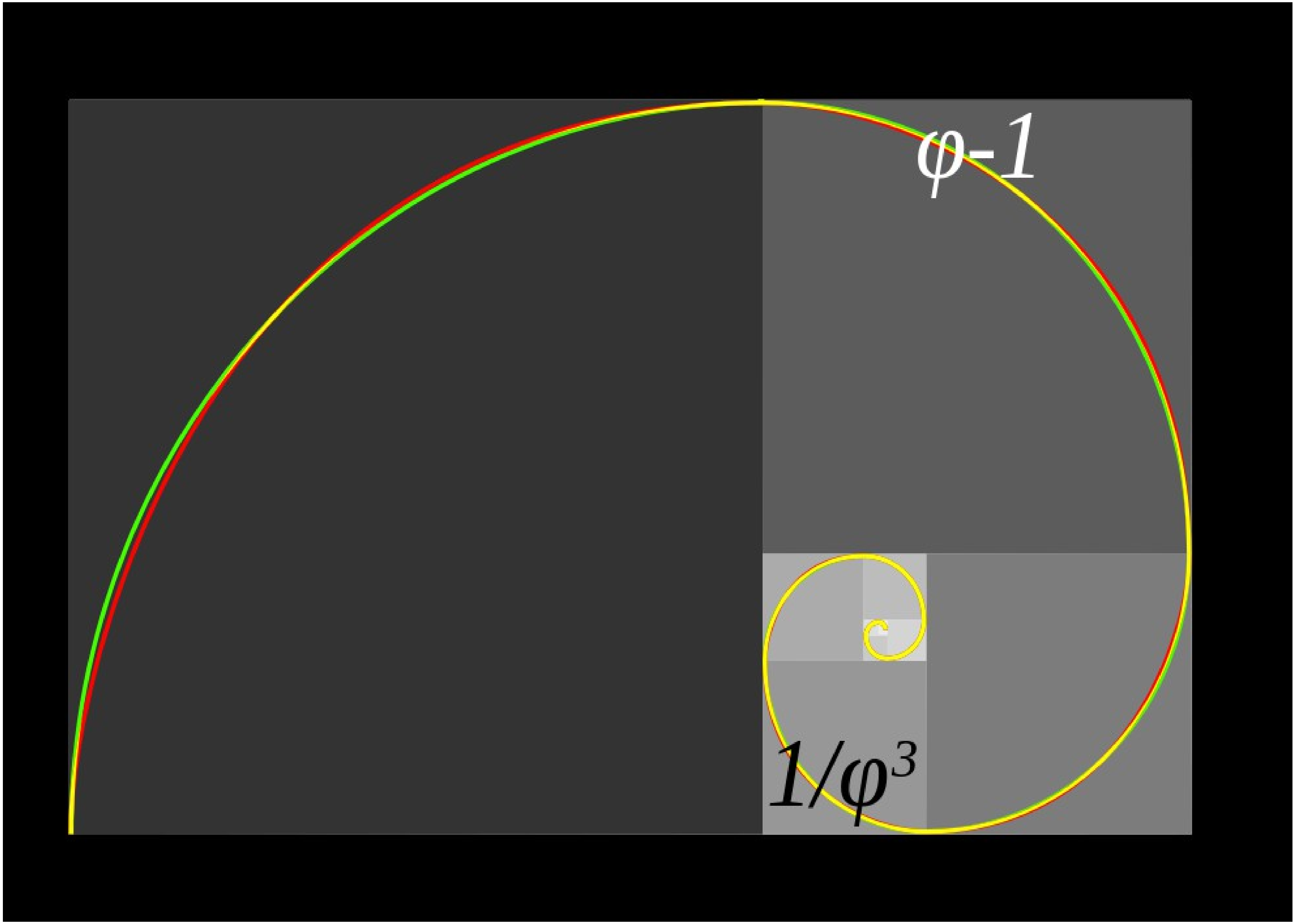
- Modules, playbooks, roles, environments?
- How can we reuse designs?
- When do I need to push or pull?
- What tool scales better? Parallelization ?
- Files (old) vs. api's (future) ?
- Why are we doing this again?
- When is it worthwhile? I just wanted to deploy 1 software package!
- Will all software/computing be SaaS ?
- IoT?

A call for Design Patterns !

“Each pattern describes a problem that occurs over and over again in our environment, and then describes the core of the solution to that problem, in such a way that you can use this solution a million times over, without ever doing it the same way twice.”

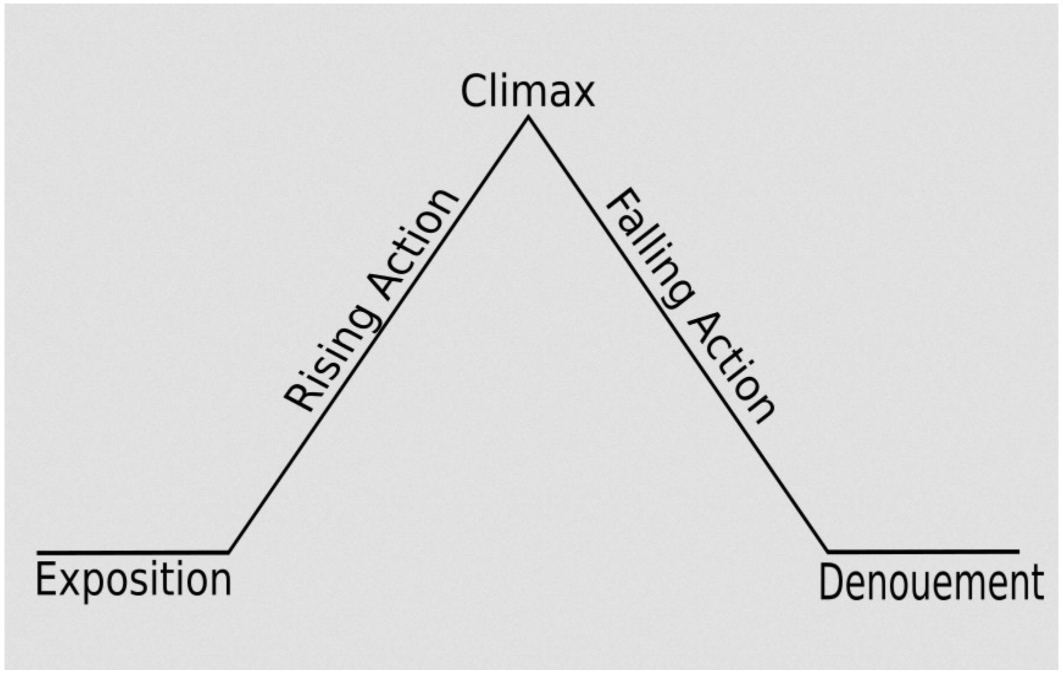
Christopher Wolfgang Alexander

Architecture

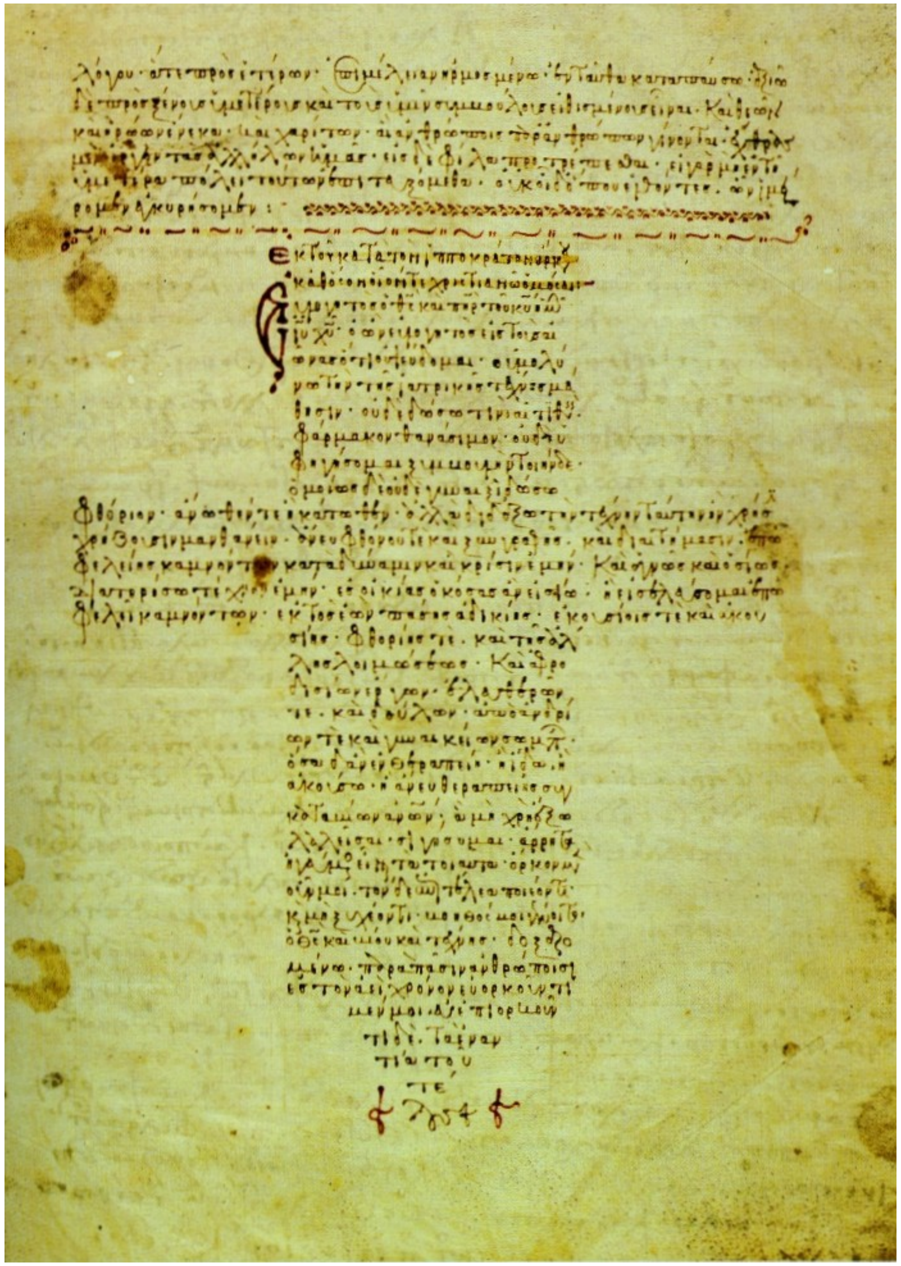

φ-1
1/φ³

Books

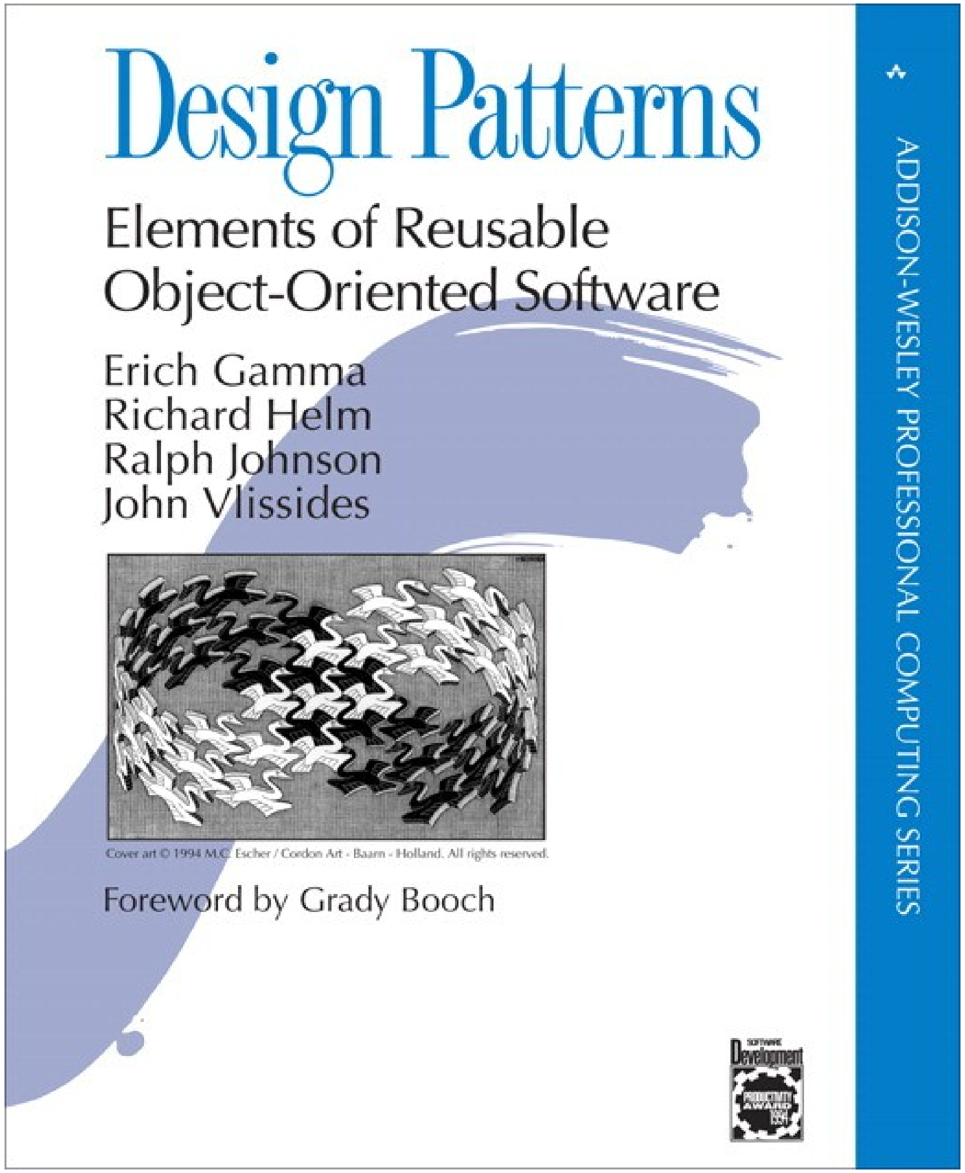
1) Overcoming the Monster
2) Rags to Riches
3) The Quest
4) Voyage and Return
5) Comedy
6) Tragedy
7) Rebirth



Medicine


[illegible]

Gang of four


Design Patterns
Elements of Reusable Object-Oriented Software
Erich Gamma
Richard Helm
Ralph Johnson
John Vlissides
Cover art © 1994 M.C. Escher / Cordon Art - Baarn - Holland. All rights reserved.
Foreword by Grady Booch
ADDISON-WESLEY PROFESSIONAL COMPUTING SERIES

Simple example of a pattern

Name : ChocolateChipRatio

Context : You are baking chocolate chip cookies in small batches for family and friends.

Consider these patterns first : SugarRatio, FlourRatio, EggRatio

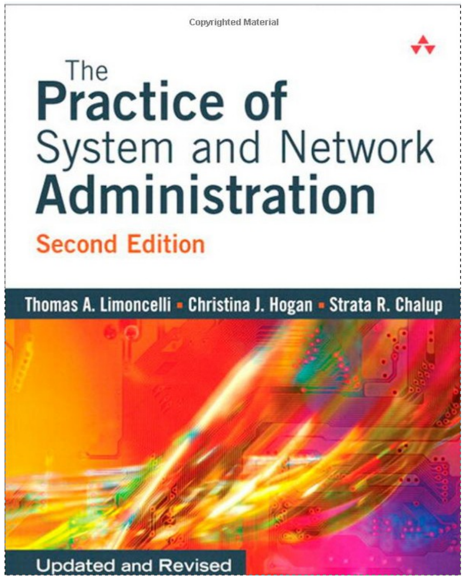
Problem : Determine the optimum ratio of chocolate chips to cookie dough.

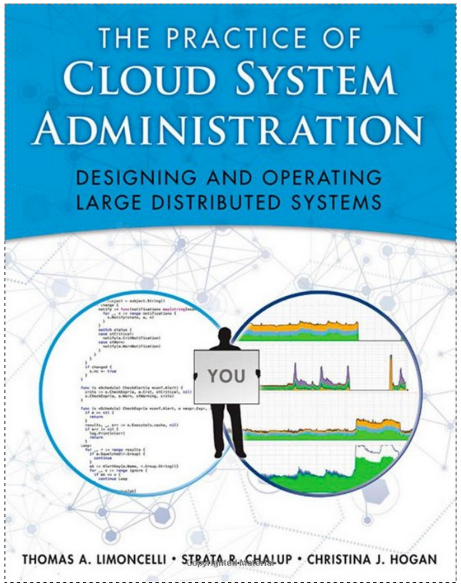
Solution : Observe that most people consider chocolate to be the best part of the chocolate chip cookie. Also observe that too much chocolate may prevent the cookie from holding together, decreasing its appeal.

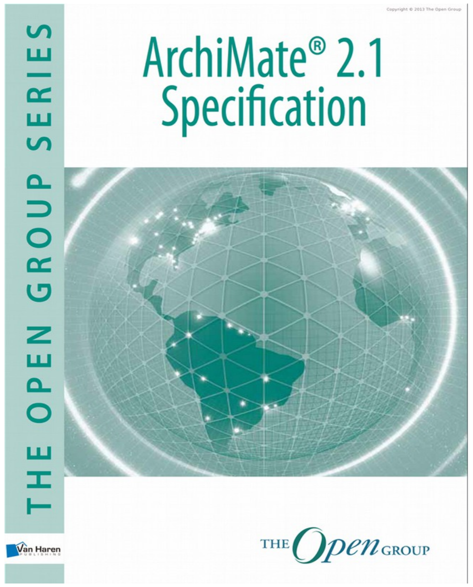
Again...

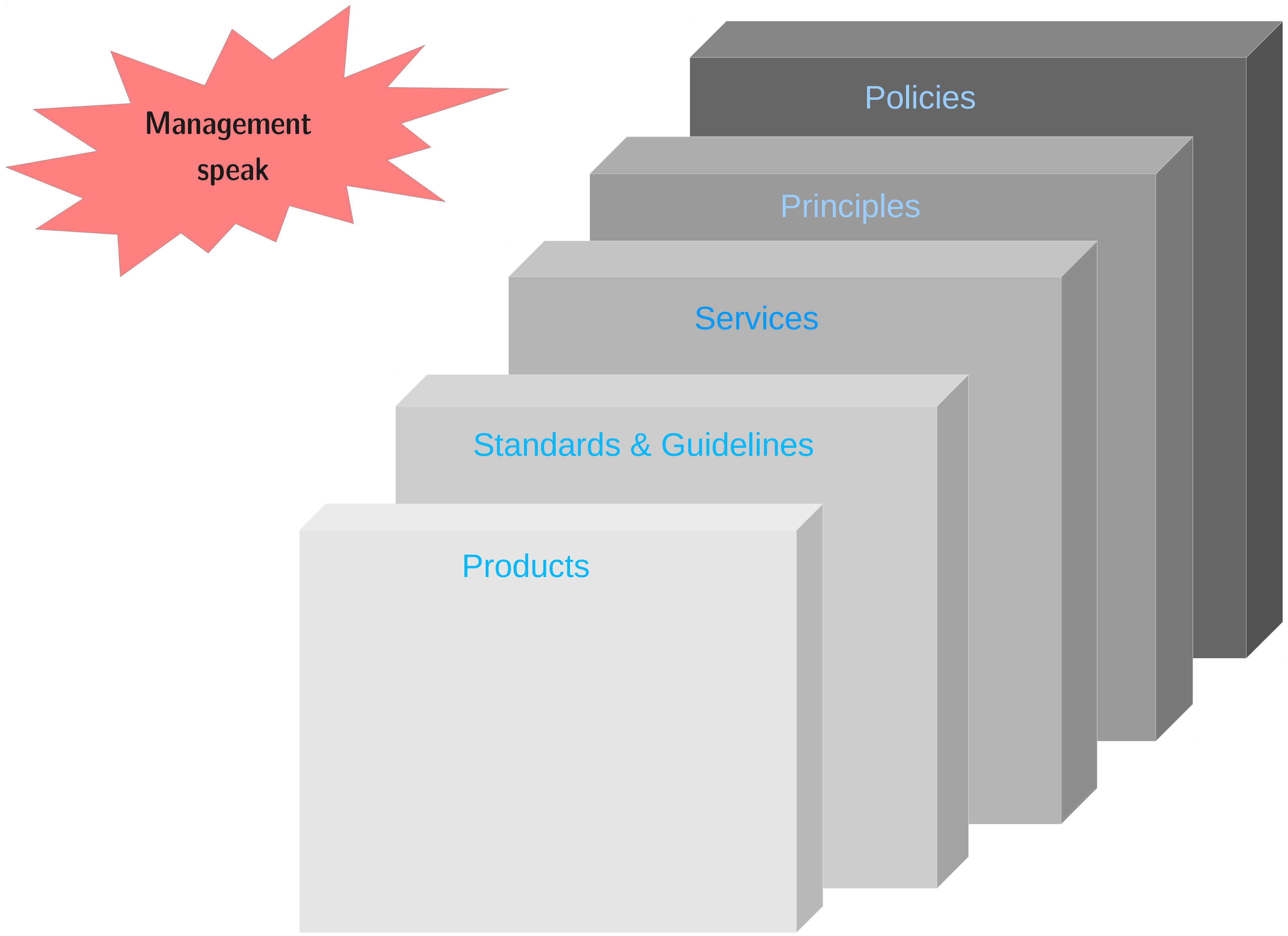
"I have yet to see these patterns codified in any meaningful way in a single work, or perhaps, an organized volume of works"

– **Brian K. Jones** Sunday, August 3rd, ***<u>2008</u>***


The Practice of System and Network Administration
Second Edition
Thomas A. Limoncelli • Christina J. Hogan • Strata R. Chalup
Updated and Revised

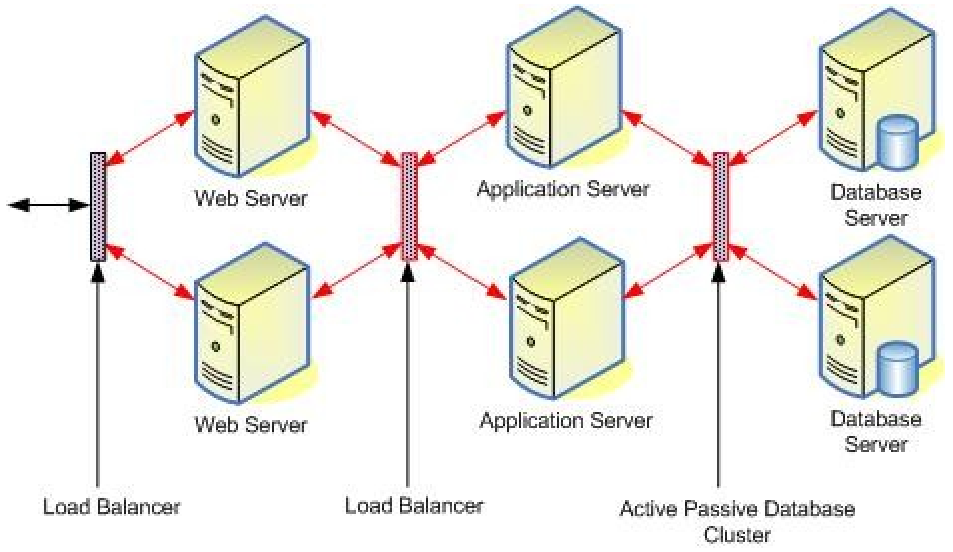

The Practice of Cloud System Administration
Designing and Operating Large Distributed Systems
YOU
Thomas A. Limoncelli • Strata R. Chalup • Christina J. Hogan

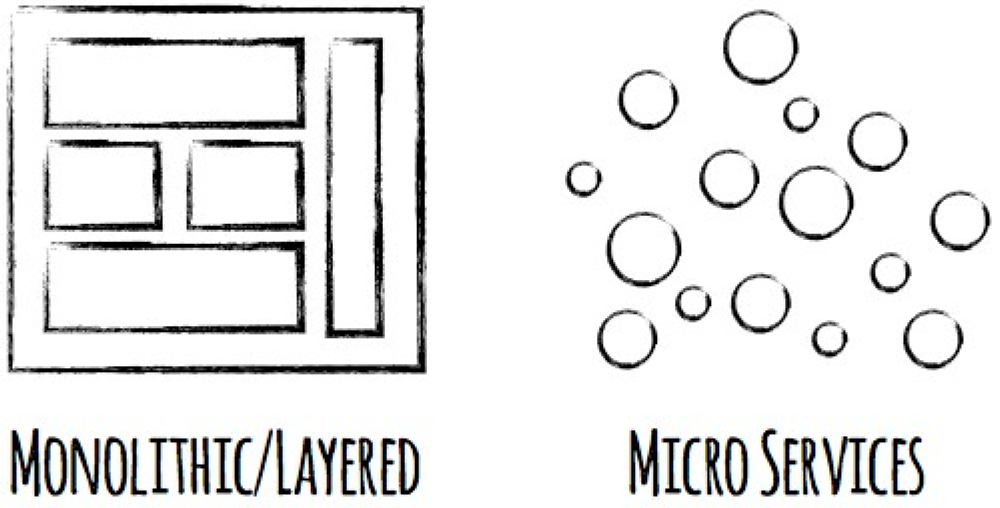



Management speak
Policies
Principles
Services
Standards & Guidelines
Products

Designs

- Multi-tier
- Micro-services
- SOA
- Distributed
- Grid





Things we just do

- Load balancing
- Partitioning
  - Vertical
  - Horizontal
- Queuing & batch
- Automate provisioning, configuration and code deploy
- Orchestration
- Golden image
- Minimize distribution of state
- Separation of concerns
- Redundancy
- Separate environments
- Monitoring
- Centralized logging

Discussion: is it useful ?

Given the future of system administration:

- “virtual” cross-datacenter networks
  - Weave, SocketPlane
- API's instead of files
  - Etcd, Consul
- Containers
  - Docker
- Simple and abstract operating systems
  - Project Atomic, CoreOS

Thank you

maurice@atcomputing.nl

Acknowledgements

Nelson Resende (FG+SG fotografia de arquitectura)

Wikipedia

Jesper Söderlund

PWC

Ordina

David A. McAfee

Brian Jones